Conditional Switching and Capacity in Neural Networks

1. The Core Idea

A purely linear system—no matter how many matrices you stack—collapses into a **single linear transformation**. This fundamentally limits its ability to store independent input–output associations.

However, if the system can **conditionally switch between different transformations based on the input**, its effective capacity grows **combinatorially**, because different inputs can follow different computational paths.

---

2. Linear Composition: No Capacity Gain

Consider two stages of linear mappings:

- First stage: A_1 or B_1
- Second stage: A_2 or B_2

If we **fix** the matrices (no switching), then a composition like:

$B_2A_2B_1A_1$

is just:

$W=B_2A_2B_1A_1$

This is a **single matrix**.

Key consequence:

- The system behaves like one linear map.
- It can only store as many independent associations as a single matrix allows.
- Adding depth **does not increase associative capacity** in the linear case.

---

3. Conditional Switching: Capacity Explosion

Now suppose we introduce **data-dependent switching**:

$(A_2 \text{ or } B_2)(A_1 \text{ or } B_1)$

This yields **four possible mappings**:

- A_2A_1
- A_2B_1
- B_2A_1
- B_2B_1

Key insight:

Each input can select a **different path**, so the system is no longer one linear map—it is a **set of linear maps**.

Capacity result:

- Instead of 1 mapping → now 4 mappings
- More generally, with k binary switches → 2^k possible transformations

This means:

> The same parameters are reused in multiple combinations, allowing the system to represent many more associations.

---

4. Interpretation: Conditional Computation

This structure behaves like:

- A **switch statement** in programming
- A **mixture of experts**
- A **lookup table of linear maps**, indexed by input-dependent decisions

Each path corresponds to a different **"mode"** of the system.

---

5. Connection to ReLU Neural Networks

ReLU networks naturally implement this kind of switching.

A ReLU activation:

$ReLU(x)=max(0,x)$

can be interpreted as a **binary gate**:

- ON (pass signal) if $x>0$
- OFF (block signal) if $x\leq 0$

---

6. ReLU as a Switching System

A ReLU layer can be written as:

$y=WD(x)x$

where:

- W is the weight matrix
- D(x) is a **diagonal matrix of 0/1 switches**, determined by the input

Each input produces a different D(x), so:

$y=WD(x)x = W_{eff}(x)x$

---

7. Exponential Number of Linear Regions

Across multiple layers:

$y=W_L D_{L-1} W_{L-1} \cdots D_1 W_1 x$

Each $D_i(x)$ introduces switching (column selection in W_{i+1}).

Result:

- The network implements **many different effective matrices**
- Each corresponds to a **region of input space**
- Number of regions grows **exponentially with depth**

---

8. Capacity Interpretation

This is directly analogous to your matrix example:

Concept	Matrix Example	ReLU Network
Fixed composition	Single matrix	Single linear region
Switching	Choose A or B	ReLU gating (on/off)
Number of mappings	2^k	Exponential in neurons/layers
Capacity	Multiplies	Explodes

---

9. Key Insight

> **Depth alone does not increase capacity. Conditional switching does.**

ReLU networks are powerful not because they stack linear maps, but because they:

- **Select different subsets of those maps**
- **Reuse parameters across exponentially many configurations**
- **Partition input space into many regions**, each with its own linear behavior